AFRL-IF-RS-TR-2006-255
Final Technical Report
July 2006

POLYMORPHOUS COMPUTER ARCHITECTURE (PCA) TECHNOLOGY TRANSITION TO JOINT SEMI AUTOMATED FORCES (JSAF)

USC Information Sciences Institute

Sponsored by
Defense Advanced Research Projects Agency
DARPA Order No. S109/00/01/02

APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED.

The views and conclusions contained in this document are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of the Defense Advanced Research Projects Agency or the U.S. Government.

AIR FORCE RESEARCH LABORATORY
INFORMATION DIRECTORATE
ROME RESEARCH SITE
ROME, NEW YORK

STINFO FINAL REPORT

This report has been reviewed by the Air Force Research Laboratory, Information Directorate, Public Affairs Office (IFOIPA) and is releasable to the National Technical Information Service (NTIS). At NTIS it will be releasable to the general public, including foreign nations.

AFRL-IF-RS-TR-2006-255 has been reviewed and is approved for publication

APPROVED: /s/

DUANE GILMOUR
Project Engineer

FOR THE DIRECTOR: /s/

JAMES A. COLLINS
Deputy Chief, Advanced Computing Division
Information Directorate

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. REPORT DATE *(DD-MM-YYYY)* JUL 06	**2. REPORT TYPE** Final	**3. DATES COVERED** *(From - To)* Apr 04 – Jan 06

4. TITLE AND SUBTITLE
POLYMORPHOUS COMPUTER ARCHITECTURE (PCA) TECHNOLOGY TRANSITION TO JOINT SEMI AUTOMATED FORCES (JSAF)

5a. CONTRACT NUMBER
FA8750-04-1-0186

5b. GRANT NUMBER

5c. PROGRAM ELEMENT NUMBER
62712E

6. AUTHOR(S)
Robert F. Lucas, Gene Wagenbreth, Dan M. Davis

5d. PROJECT NUMBER
S109

5e. TASK NUMBER
SC

5f. WORK UNIT NUMBER
TC

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
USC Information Sciences Institute
4676 Admiralty Way
Marina Del Ray California A 90292-6695

8. PERFORMING ORGANIZATION REPORT NUMBER
N/A

9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)

Defense Advanced Research Projects Agency
3701 North Fairfax Drive
Arlington Virginia 22203-1714

AFRL/IFTC
525 Brooks Road
Rome New York 13441-4505

10. SPONSOR/MONITOR'S ACRONYM(S)

11. SPONSORING/MONITORING AGENCY REPORT NUMBER
AFRL-IF-RS-TR-2006-255

12. DISTRIBUTION AVAILABILITY STATEMENT
APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED. PA#06-552

13. SUPPLEMENTARY NOTES

14. ABSTRACT
This effort investigated the feasibility of applying the University of North Carolina's OneSAF Objective System Graphics Processing Unit (GPU) algorithms to the Joint Semi Automated Forces (JSAF) code, and its civilian derivative, Culture. These codes are primarily used for situational understanding in the U.S. Joint Forces Commands (JFCOM) Urban Resolve experiments. Urban Resolve experiments include urban battle spaces, red and blue forces, a broad mix of sensor platforms, and very large numbers of civilian entities. Line-of-sight (LOS) and route planning calculations are both extensively used, and this effort explored the opportunity of exploiting GPUs to accelerate them. It was determined that both LOS and route planning can be significant performance bottlenecks in JSAF. They are not such bottlenecks that order-of-magnitude improvements can be hoped for. However, investigation determined that factors of two improvements in computational speed are feasible. Given the relatively low cost of GPUs when compared to commodity Linux computing systems, they would represent a cost-effective system upgrade for Joint Experimentation. As a result of this research project, JFCOM has submitted a proposal to the DoD High Performance Computing Modernization Program for a GPU-enhanced cluster.

15. SUBJECT TERMS
Polymorphous Computing Architecture, Large-scale simulations, high-performance computing, Graphics Processing Units, line of sight, route planning

16. SECURITY CLASSIFICATION OF:			**17. LIMITATION OF ABSTRACT**	**18. NUMBER OF PAGES**	**19a. NAME OF RESPONSIBLE PERSON** Duane Gilmour
a. REPORT U	**b. ABSTRACT** U	**c. THIS PAGE** U	UL	20	**19b. TELEPONE NUMBER (*Include area code*)**

Table of Contents

List of Figures

Acknowledgements

This material is based on research sponsored by AFRL and DARPA under agreement number FA8750-04-1-0186. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon.

The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of AFRL and DARPA or the U.S. Government.

1.0 Executive Summary

This is the Final Performance Report for the Polymorphous Computer Architecture (PCA) Technology Transition to the Joint Semi Automated Forces (JSAF) (PCA Tech Tran) project being performed for the Air Force Research Laboratory, Information Directorate. It covers the period from 01 April 2004 through 19 January 2006.

PCA Tech Tran was a project that was created to explore technology transfer between the Defense Advanced Research Projects Agency (DARPA) PCA program and the Joint Forces Command (JFCOM) Joint Experimentation (JE) community.

DARPA and the US Army's Research and Development Command (RDECOM) contracted with the University of North Carolina (UNC) to explore the possibility of using PCA software technology to accelerate the performance of the U.S. Army's OneSAF Objective System (OOS) code using Graphic Processing Units (GPUs). UNC, together with its subcontractors Scientific Applications International Corp. (SAIC) and Stanford University, determined that three computational bottlenecks of OOS suitable for exploiting GPUs were: line-of-sight (LOS) determination; route planning; and collision detection.

The PCA Tech Tran project explored the possibility of transferring this same PCA technology to JFCOM's Joint Experimentation community. The Information Sciences Institute (ISI) of the University of Southern California (USC) looked into the feasibility of exploiting UNC's OOS GPU algorithms in JFCOM/J9's Urban Resolve experiments. LOS and route planning are both extensively used in Urban Resolve, and ISI analyzed their performance impact and hence the opportunity of exploiting GPUs to accelerate them. JFCOM decided that collision detection amongst vehicles on the ground was currently not of vital concern in Urban Resolve, and thus ISI did not examine this.

ISI found that both LOS and route planning can be significant performance bottlenecks in the JSAF code, as employed by JFCOM in Urban Resolve. They are not such bottlenecks that order-of-magnitude improvements can be expected. However, factors of two improvements are feasible. Given the relative cost of commodity Linux computing systems, versus GPUs, this still represents a cost-effective system upgrade and hence a suitable opportunity for PCA technology transition. Therefore, based upon the results of this research project, JFCOM submitted a 2006 Dedicated High performance computing Project Investment (DHPI) request to the DoD High Performance Computing Modernization Program (HPCMP) for a GPU enhanced PC cluster.

2.0 Introduction

It is believed that streaming language and compiler technology from the DARPA PCA program may enable current Defense applications to exploit the multi-media extensions of modern micro-processors (e.g., Intel's SSE or PowerPC's Altivec) as well as GPU coprocessors. This would be an example of opportunistic technology transfer, by deploying streaming programming technology long before its intended use in future, polymorphic computing systems. In particular, DARPA and the US Army's RDECOM contracted with UNC to explore the possibility of using PCA software technology to accelerate the performance of the U.S. Army's OOS code using GPUs. UNC, together with its subcontractors SAIC and Stanford University, determined that three computational bottlenecks of OOS suitable for exploiting GPUs are: line-of-sight determination (LOS); route planning; and collision detection.

ISI is a major research institute concentrating on computer and network applications and systems for the Department of Defense (DoD). ISI has had a sequence of projects (MOrphable Networked microARCHitecture (MONARCH), MCHIP, and XMONARCH) that support DARPA's PCA program [Granacki 2004]. Personnel from ISI's Computational Sciences Division played a key role in these efforts, which included early research into streaming languages and compilers for polymorphic systems. ISI personnel are also actively involved in supporting the US JFCOM's Joint Experimentation Directorate (J9). In the Joint Experimentation on Scalable Parallel Processor Computers (JESPP) project, ISI has been expanding the horizons of JFCOM's JSAF [Ceranowicz 2002] code for use at ever-increasing scale and sophistication [Lucas 2003 and Wagenbreth 2005]. The JESPP project represents transition of earlier DARPA research results [Messina 1997] to JFCOM, including a new communication architecture [Gottschalk 2005]. Both OOS and JSAF evolved from the same Modular Semi Automated Forces (ModSAF) code base and much of their software architecture reflects that heritage. Thus, ISI was ideally situated to explore the possibility of transitioning PCA technology targeted at OOS to the joint experimentation community and its JSAF code.

3.0 Methods, Assumptions and Procedures

ISI looked into the feasibility of exploiting UNC's OOS GPU algorithms in JFCOM/J9's JSAF code, and its civilian derivative, Culture. These codes are primarily used for situational understanding in JFCOM/J9's Urban Resolve experiments [Ceranowicz 2005]. Urban Resolve experiments include urban battle spaces, red and blue forces, a broad mix of sensor platforms, and very large numbers of civilian entities. LOS and route planning are both extensively used, and ISI explored the opportunity of exploiting GPUs to accelerate them. J9 has determined that collision detection amongst vehicles on the ground is not a current priority for Urban Resolve, and thus ISI did not examine this.

As will be discussed in this report, ISI found that both LOS and route planning can be significant performance bottlenecks in JSAF, and its civilian derivative, Culture, as employed by JFCOM in Urban Resolve. They are not such bottlenecks that order-of-magnitude improvements can be hoped for. However, ISI's investigation determined that factors of two improvements in computational speed are feasible. Given the relatively low cost of GPUs when compared to commodity Linux computing systems, they would represent a cost-effective system upgrade for Joint Experimentation.

These results have been presented to DARPA and RDECOM at UNC project reviews, to AFRL in quarterly reports, and have also been communicated privately to JFCOM/J9. Broader dissemination has also been proposed via an abstract submitted to the next Interservice/Industry

Training, Education and Simulation Conference (I/ITSEC 2006). The abstract, which was accepted, is attached in Appendix A.

3.1 Culling for Line-of-Sight

This section begins with a brief review of the LOS problem and UNC's algorithm for reducing this bottleneck by culling on GPUs. Next is a description of the LOS bottleneck as it occurs in Urban Resolve. This is followed by a discussion of a series of experiments ISI conducted to determine the impact of LOS on JSAF, and hence the magnitude of the opportunity for exploiting the UNC GPU algorithm. Finally the results are presented.

In open battlefields, or other scenarios where interest filtering is not possible, LOS is an $O(N^2)$ problem, where N is the number of entities simulated. Thus, as the scale and complexity of military training and experimentation increase, the time to determine whether or not entities can see each other can quickly become the computational bottleneck. The most costly LOS queries are between remote entities, as the processor has to traverse the entire terrain surface between the two entities, testing to see whether the terrain itself, or any object on it, obstructs the line-of-sight.

UNC developed a hybrid GPU/CPU algorithm which performs conservative culling in the GPU portion of the algorithm. LOS queries whose segments are definitely unblocked are quickly culled away by the GPU, thereby reducing the number of LOS queries that must be tested exactly by the CPU. Queries with unblocked line of sight are the most expensive for the CPU and many of these are culled. They can thus become the best, rather than the worst, performance case if culled by the GPU. UNC results for open battlefields in OOS demonstrate that 10X speedups are possible [Salomon 2004].

3.1.1 Line-of-Sight in JSAF and Urban Resolve

JFCOM's Urban Resolve experiments differ significantly from the OOS scenarios studied by UNC. First, they are primarily conducted in urban terrain, which is much more complicated than open terrain, and where line-of-sight is unlikely to exist between remote entities. Second, most of the simulated entities are simple civilian pedestrians and vehicles which are simpler and more numerous than the high fidelity military vehicles simulated by OOS in UNC's experiments. The civilian entities are collectively called "culture" and are simulated by a program called Culture, which is descended from JSAF. At one time culture entities were called "clutter". Where the term "clutter" appears in this text it is synonymous with "culture". Culture entities do not use LOS to detect each other and/or avoid collisions. Separate, highly efficient intersection logic performs this function. Each CPU typically simulates several thousand culture entities. JSAF and OOS typically simulate hundreds, rather than thousands, of full-fidelity military entities. The principal source of LOS calculations in Urban Resolve is collections of sensors, including satellites, whose sensor footprints cover areas of interest and detect the entities therein. The sensors are simulated by a program named Simulation of the Locations and Attack of Mobile Enemy Missiles (SLAMEM). Urban Resolve experiments used hundreds of CPUs to simulate culture. To efficiently perform the LOS algorithm in a distributed memory environment, the program simulating an entity that may have been illuminated, rather than SLAMEM, performs the LOS calculation to see if the sensor can see said entity.

3.1.2 Line-of-Sight Experiments

ISI instrumented the JSAF code and performed experiments to measure the amount of time taken when performing LOS calculations. Both open and urban terrain scenarios were examined. Scenarios with tanks as well as scenarios with culture and satellite sensors were used. The open terrain scenarios required each of the simulated tanks to perform LOS calculations to all other tanks. The urban scenarios required each culture entity to perform a LOS calculation to each sensor when in the sensor's footprint. These latter scenarios are representative of the common use of JSAF and the Culture simulator in Urban Resolve

For each scenario, statistics were gathered and logged every 10 seconds. The following information was gathered for each 10 second period:

- wall clock time (10 seconds)
- cpu time for non LOS calculations
- cpu time for LOS calculations with blocked LOS
- cpu time for LOS calculations with unblocked LOS

Code was inserted in the routine ctdb_point_to_point() in the file libctdb/ct_ptop.c. Timings were written to stdout at execution time. A program was run to read the statistics from the output file and write them to a data file in a format suitable for the program Gnuplot. Gnuplot was then used to generate plots. The results appear in Figures 1 through 4.

3.1.3 Line-of-Sight Results and Discussion

Each plot in Figures 1 through 4 is a series of bars, each representing a 10 second time period. The vertical scale is percentage of CPU time, 0 - 100. The horizontal scale is time. The height of the bar is the percentage of CPU time used by the simulator. If the bar is full scale, i.e. 100%, the CPU is saturated. Each bar is divided into three colors. Red is the percentage of CPU time used by non-LOS calculations. Green is the percentage of CPU time used by LOS calculations that are blocked. Blue is the percentage of CPU time used by LOS calculations that are unblocked.

Figure 1 shows results for 120 tanks performing LOS calculations between each other in open terrain. LOS calculations consume less than five percent of the CPU time. There are few obstacles or terrain features to examine to determine visibility. The simulation of 120 tanks does not saturate the CPU.

Figure 2 shows results for 120 tanks performing LOS calculation between each other in urban terrain. LOS calculations use twenty percent of the CPU time. The result is almost always that

LOS is blocked, as is expected for tanks moving between buildings. The CPU is saturated. Both LOS and non-LOS calculations are more time consuming in an urban environment.

Figures 3 and 4 show results for 2000 culture entities scanned by satellites. Eight satellites, equally spaced in a common orbit, were simulated. When a satellite comes over the horizon its footprint covers the location of the entities and many LOS calculations are performed. When it passes back over the horizon the LOS calculations terminate until the next satellite appears. Figure 3 shows timings in an open environment. Figure 4 shows timings in an urban environment. Both Figures show spikes as satellites pass overhead. In the open environment, LOS is almost always unblocked. In an urban environment some LOS is blocked. Both LOS and non-LOS CPU usage is higher in the urban environment.

The data from 120 tanks in an open environment shows that LOS calculations, as performed by JSAF, are a relatively insignificant portion of the CPU time. In an urban environment, the LOS calculations amongst these same 120 tanks take closer to 20% of the CPU usage. The $O(N^2)$ complexity of LOS suggests that for large numbers of entities, simulations of urban operations could be dominated by blocked LOS calculations. Experience to date with Urban Resolve suggests that the limited number of operation entities, together with interest filtering, keeps this potential bottleneck to a manageable level.

Scanning of culture entities by sensors, the scenarios depicted in Figures 3 and 4, is ubiquitous in Urban Resolve. The data show that in this mode LOS calculations are made in bursts when sensor footprints overlap large numbers of culture entities. Between these bursts, LOS uses no CPU time. When a sensor with a large footprint moves over culture entities, LOS calculations can consume over 50% of the available CPU time. The CPU often becomes saturated, temporarily limiting the update rate of the entities. This is a significant problem for experiments that are intended to progress in real time.

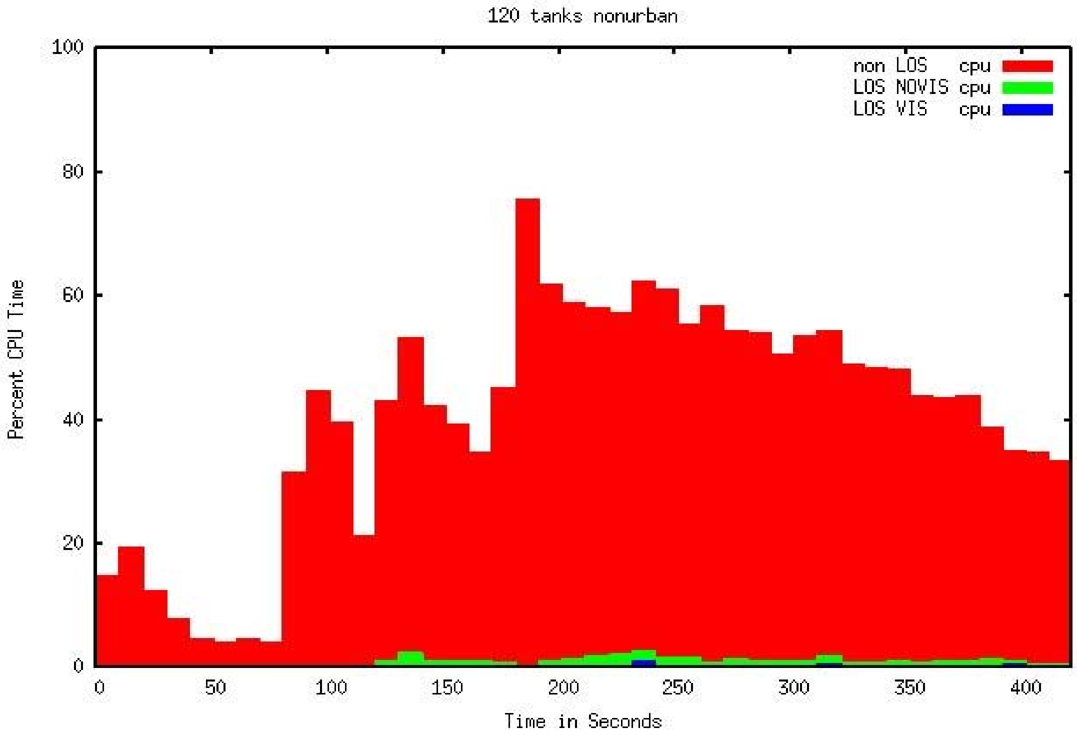

Figure 1 - 120 Tanks in Open Terrain

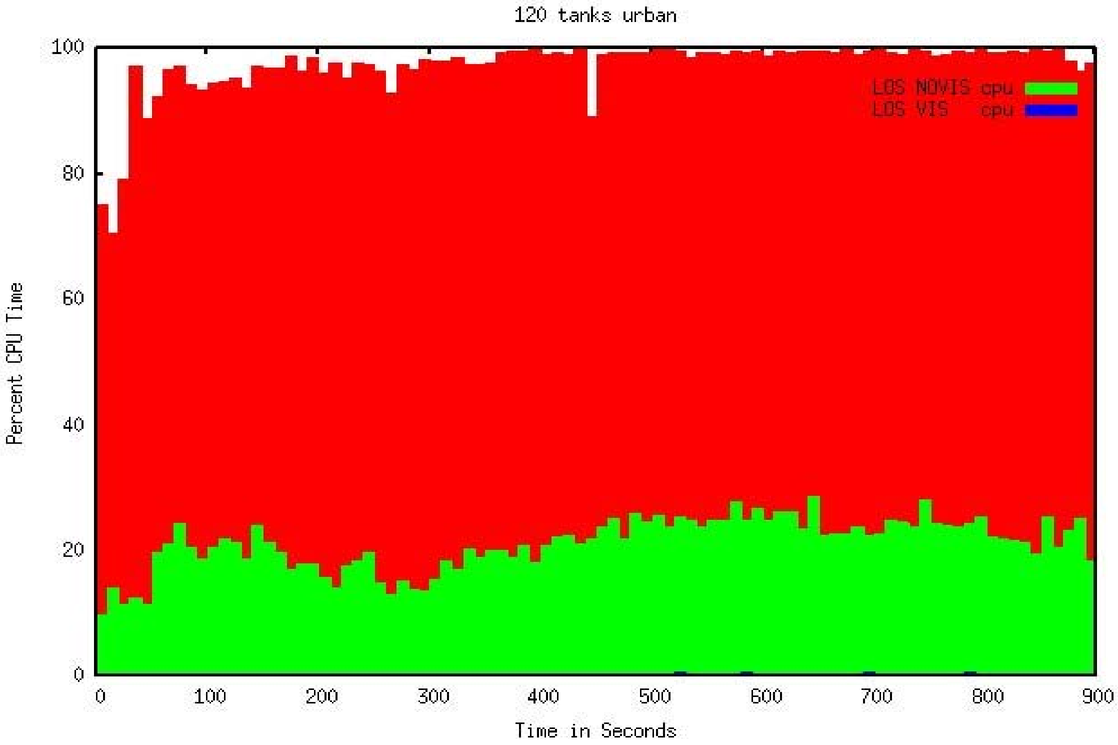

Figure 2 - 120 Tanks in Urban Terrain

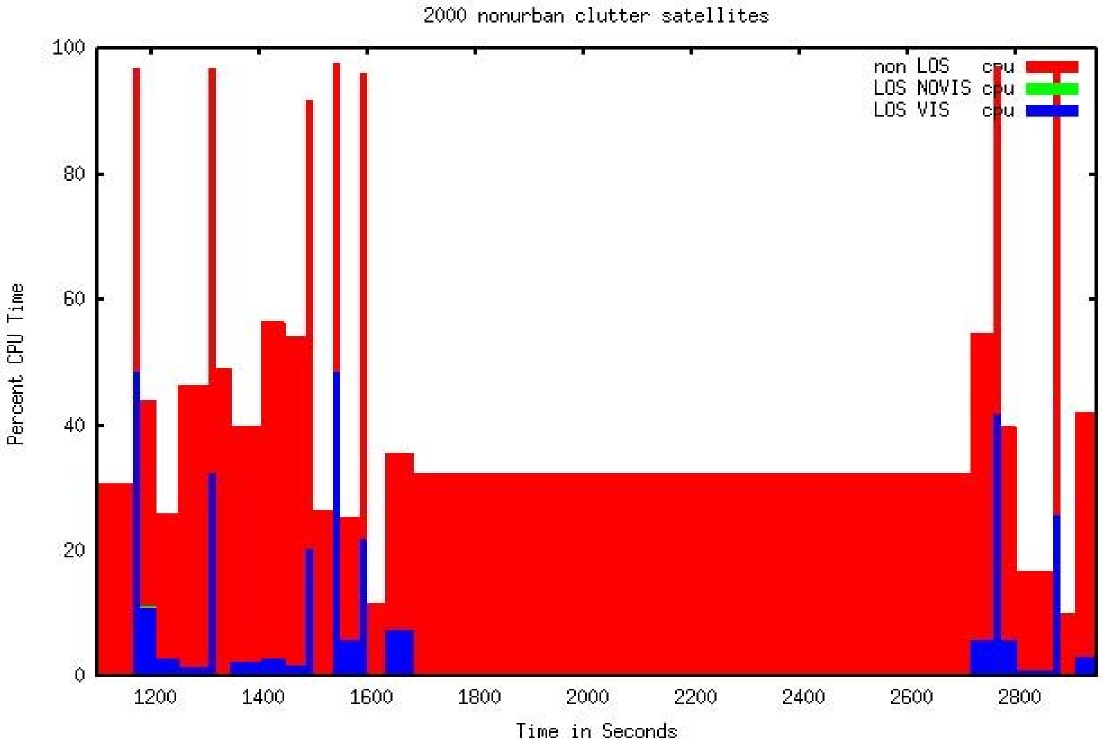

Figure 3 - 2000 Culture Entities in Open Terrain with Satellite Sensors

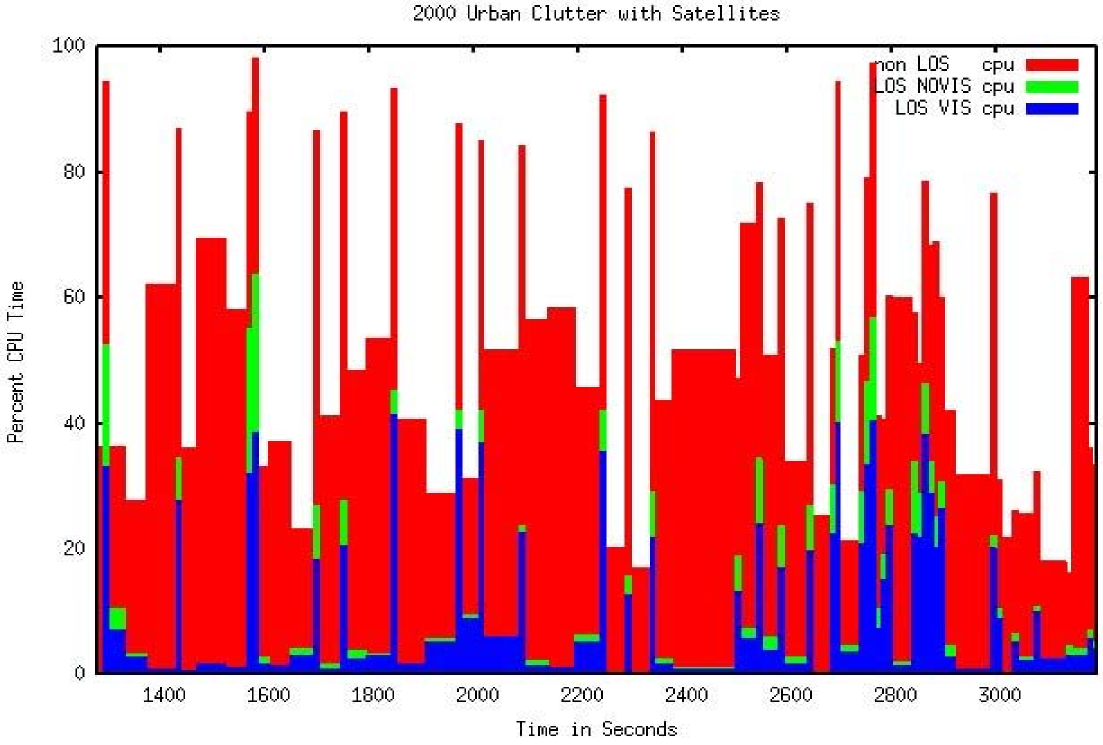

Figure 4 - 2000 Culture Entities in Urban Terrain with Satellite Sensors

3.2 Culling for No Line-of-Sight

The UNC GPU LOS culling algorithm was designed to rapidly detect cases where line-of-sight exists between two remote entities, obviating the need for the host processor to do this expensive calculation itself. As shown above, in urban terrain, the vast majority of LOS queries fail. Thus one is tempted to instead consider the use of GPUs to cull for the case where no line-of-sight exists.

Due to the lack of memory and other resources in the GPU, it is not possible to represent the terrain on the GPU with full fidelity. The UNC culling algorithm approximates terrain with circumscribed polygons. This allows efficient execution of the LOS algorithm. Because the polygons are circumscribed, a result of "can see" is accurate, whereas a result of "can not see" may not be. To be sure, the "can not see" LOS calculation must be recomputed exactly by the CPU. If, as in Urban Resolve, most results are "can not see", the GPU LOS culling does not speed up the code.

It may be possible to use a GPU efficiently in urban terrain by using inscribed, rather than circumscribed polygons. If a terrain feature such as a building is approximated by one or more polygons entirely in its interior, the UNC culling algorithm is reversed. A "can not see" result is accurate. A "can see" result is approximate and must be verified by the CPU. The rectilinear geometry of most buildings and other human artifacts in an urban environment may make this possible. As of this writing, it is unknown whether or not openings such as windows and doorways will require the use of too many polygons for the inscribed polygons to work efficiently. Nevertheless, we believe this approach warrants further study using detailed information from urban terrain files. If the approach works, the circumscribing algorithm can be combined with the inscribed algorithm:

```
if(circumscribed_algorithm() == CANSEE) return CANSEE
if(inscribed_algorithm() == CANNOTSEE) return CANNOTSEE
return exact_algorithm()
```

A diagram illustrating the use of inscribed and circumscribed terrain approximations is shown in Figure 5. Two sensors, one on a satellite and one on a helicopter, are shown. The potential targets are the automobiles and antennae in the Figure. Two buildings and a parking garage are terrain features which may mask the targets from the sensors. The UNC culling algorithm generates the circumscribed polygons shown with dashed green lines to approximate the buildings. The algorithm proposed by ISI generates the inscribed polygons shown with dashed red lines to approximate the buildings. In both cases the polygons are only an approximation to the actual terrain. Note that from directly above the scene, the sensor on the satellite can see all of the entities in the open (i.e., all but the car in the parking garage). The sensor on the helicopter can only see a fraction of them. In the latter case, culling using the circumscribed and inscribed terrain leaves only those lines of sight that pass near the edges of the terrain for the CPU to evaluate.

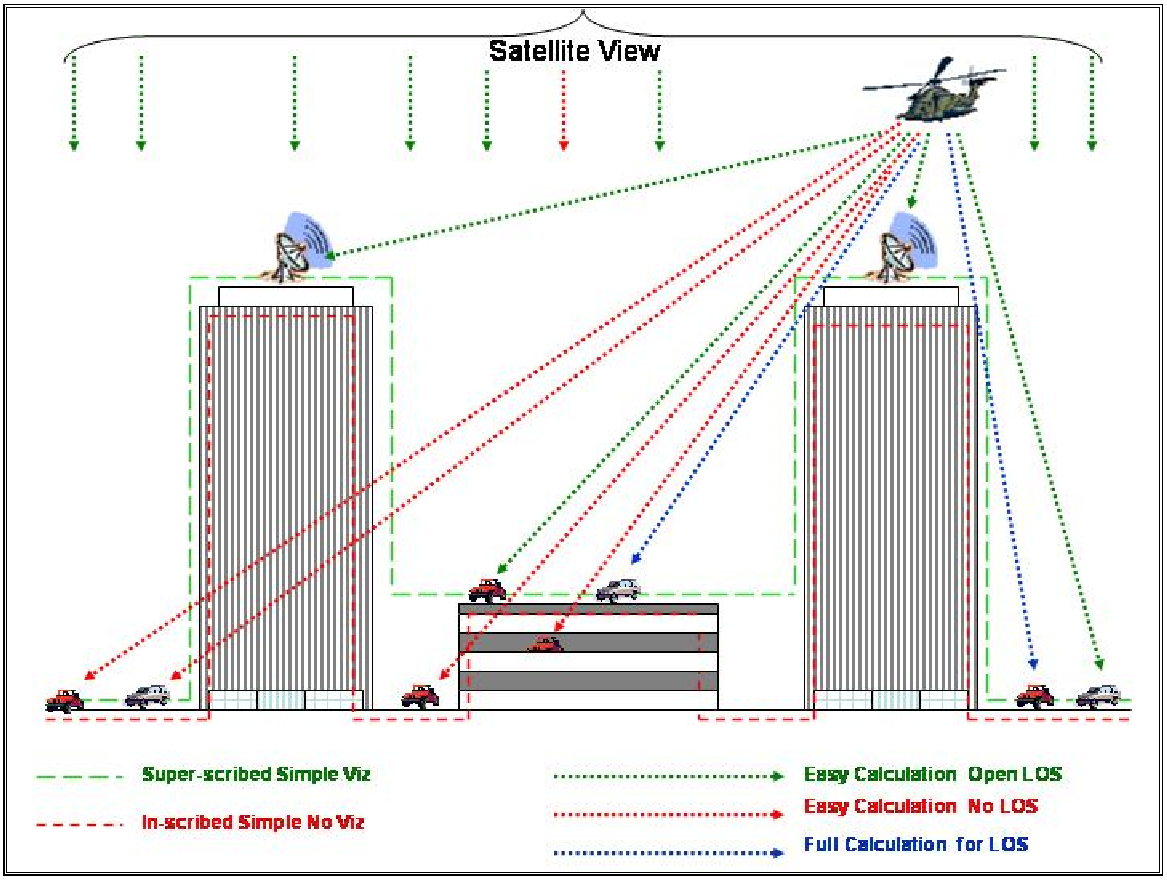

Figure 5 - Illustration of Circumscribed and Inscribed Terrain

3.3 Route Planning

This section begins with a brief review of route planning. This is followed by a discussion of a series of experiments ISI conducted to determine the impact of route planning on JSAF and culture, and hence the magnitude of the opportunity for exploiting the UNC GPU algorithm. Finally the results are presented.

JFCOM's Urban Resolve experiments involve over 1,000,000 simulated entities moving about the urban battle space. Low fidelity culture entities are assigned one of a multitude of behaviors such as:

- commuter
- delivery truck
- taxi
- police car
- soccer mom

As appropriate for the assigned behavior, each entity, depending on the time, is given a source and a destination. A commuter goes from home to work in the morning and from work back home in the evening. It may go to a restaurant in the evening. Each behavior requires moving from a source to a destination at an appropriate time. Given the source and destination, an efficient route is planned using the available road network. The road networks in Urban Resolve experiments are derived from an accurate map of a real city, such as Baghdad.

Route planning can utilize a significant portion of the available CPU time. Fortunately, it is contained in a relatively small section of code such that it can be easily studied. UNC demonstrated that route planning in OOS can be ported effectively to a GPU.

3.3.1 Route Planning Experiment

To determine what impact a GPU implementation of route planning might have on Urban Resolve, ISI instrumented the JSAF code and performed experiments to determine how much CPU time is consumed in route planning. It was found that route planning is most intense when culture entities are first created. At this time, all of the new entities immediately begin planning their first route. CPU usage eventually evens out as these new entities start their trips at uniformly distributed times. Therefore, in each experiment, groups of entities were created in intervals to limit peak CPU utilization. The same procedure is used in Urban Resolve to avoid creating computational bottlenecks that cause the simulation to fail to proceed in real time.

For each scenario, statistics were gathered and logged every 10 seconds. The following information was gathered for each 10-second period:

- Wall-clock time (10 seconds)
- CPU-time for non-route planning calculations
- CPU time for route planning calculations

Code was inserted in the routine traverse_roads() in the file libclutterpath/clpath_path.c. Timings were written to stdout at execution time. A program was run to read the statistics from the output file and write them to a data file in a format suitable for Gnuplot. Gnuplot was then used to generate plots. The results appear in Figures 6 and 7.

3.3.2 Route Planning Results and Discussion

Figures 6 and 7 contain a series of bars, one for each 10 second time period. The vertical scale is percentage of CPU time, 0 - 100. The horizontal scale is time. The height of the bar is the percentage of CPU time taken by the simulator. If the bar is full scale, i.e. 100%, the CPU is saturated. The bar is divided into 2 colors. Red is the percentage of CPU time used by calculations other than route planning. Green is the percentage of CPU time used by route planning calculations.

Figure 6 shows the results for creating 12500 culture entities, 2500 at a time, while waiting 3 or 4 minutes between increments. Figure 7 shows the results for creating 18000 culture entities, 1000 at a time, and again waiting 3 or 4 minutes between increments. Both data sets show a spike in route planning CPU utilization when each increment of culture entities is created. For approximately one minute, route planning uses most of the CPU time and the CPU is saturated. After this minute, the initial burst of route planning is done and route planning drops to a sustained 10% – 30% of the total CPU time used by the simulator. The total CPU time increases as the number of simulated entities is increased. The amount of CPU time used for route planning and other calculations depends on the road network of the area containing the culture entities. These data sets come from two different areas of Jakarta. The CPU time per entity is smaller in the second dataset. The pattern of route planning and non-route planning CPU utilization is very similar.

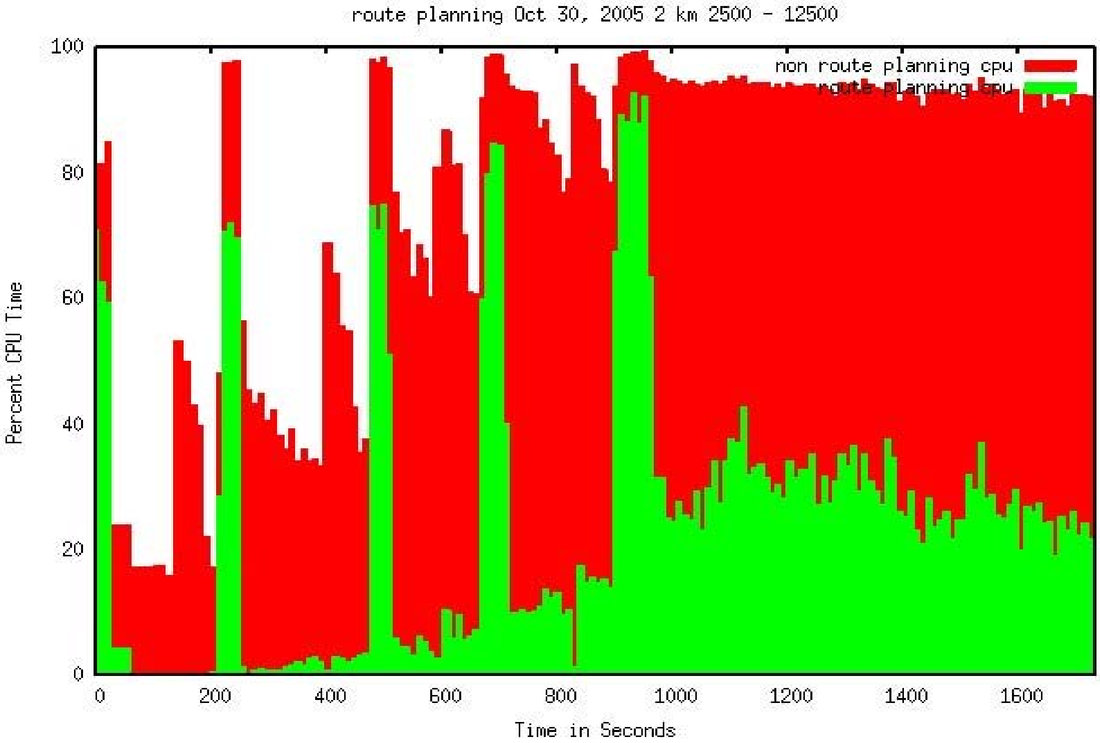

Figure 6 - Route Planning 12500 Entities in Increments of 2500

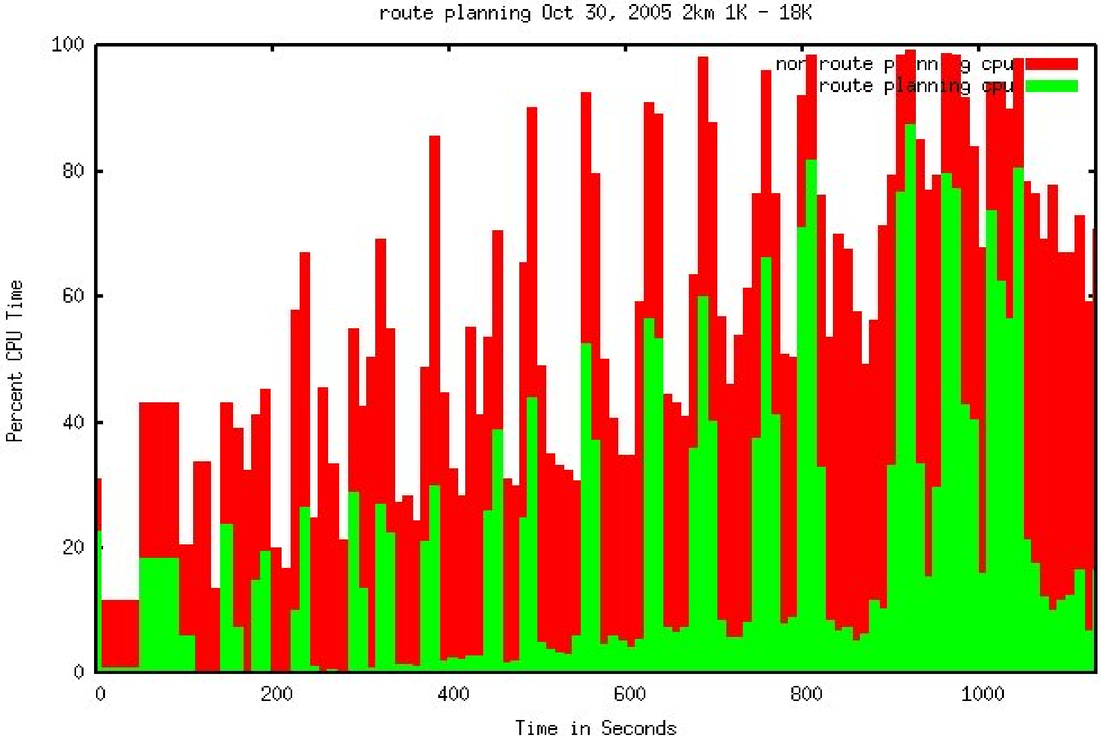

Figure 7 - Route Planning 18000 Entities in Increments of 1000

4.0 Conclusions

DARPA and RDECOM contracted with UNC to determine if PCA streaming programming technology could be used to accelerate the throughput of the Army's OOS code. Three computational bottlenecks, line-of-sight determination, route planning, and collision detection, were determined to be suitable for exploitation by GPUs. Test cases were constructed that demonstrated speedups of an order of magnitude when the GPU algorithms were used, as opposed to the baseline, Java implementations of the algorithms.

Early evidence that GPUs might prove useful in OOS led DARPA to issue a subsequent contract to ISI to examine the possibility of further transition of this technology to JFCOM's Urban Resolve experiments. The Urban Resolve experiments revolve around operations in urban terrain using the JSAF, Culture, and SLAMEM codes. ISI focused its analysis on the line-of-sight and routing planning functions, since collision detection is not a bottleneck in Urban Resolve.

ISI found that both line-of-sight and route planning calculations can be significant computational bottlenecks in Urban Resolve. At their peaks, they can consume half of the CPU time in real urban scenarios.

4.1 Recommendations

In order to ensure that experiments are able to progress in real time, Urban Resolve software engineers have to provision for the above described peaks in utilization, and thus are forced to halve the number of entities they could otherwise simulate.

ISI's analysis demonstrates that there is an opportunity for GPUs, programmed with PCA streaming language technology, to be used to address the computational peaks associated with line-of-sight calculations between culture entities and high-flying sensors. Because most line-of-sight calls fail in the urban environment, we have proposed an alternative algorithm that culls for non-line-of-sight. The UNC route planning algorithm should be able to similarly reduce the computational bottleneck associated with route planning. In both cases, the GPUs are used to "clip" the CPU utilization peaks.

Overall, factors of two in performance improvement appear to be possible. When the advance of time is constrained to real time, as it is in Urban Resolve, the CPU power freed by the use of GPUs could be used to increase the number of entities simulated. GPUs are relatively inexpensive when compared to Linux PCs, whether on desktops or in clusters. Thus they should be a very cost-effective upgrade for the systems employed in Urban Resolve. As a result of this research project, JFCOM has submitted a proposal to HPCMP's 2006 DHPI solicitation for a GPU-enhanced cluster. This represents the first step in the transition of PCA streaming technology to Joint Experimentation at JFCOM.

5.0 References

Ceranowicz, A. & Torpey, M., (2005), Adapting to Urban Warfare, *The Journal of Defense Modeling and Simulation,* Vol 2 No 1, January 2005, San Diego, California

Ceranowicz, A., Torpey, M., Hellfinstine, W., Evans, J. & Hines, J. (2002) Reflections on Building the Joint Experimental Federation. *Proceedings of the Interservice/Industry Training, Simulation and Education Conference*, Orlando, FL

Gottschalk, T., Amburn, P. &Davis, D., (2005), "Advanced Message Routing for Scalable Distributed Simulations," *The Journal of Defense Modeling and Simulation,* Vol 2 No 1, January 2005, San Diego, California

Granacki, J., “MONARCH: Next Generation SoC (Supercomputer on a Chip)”, *HPEC 2004*, Lexington, MA

Lucas, R., & Davis, D., (2003),"Joint Experimentation on Scalable Parallel Processors," in the *Proceedings of the Interservice/Industry Simulation, Training and Education Conference*, Orlando, Florida, 2003

Messina, P., Brunett, S., Davis, D., Gottschalk, T., Curkendall, D., & Seigel, H., (1997) "Distributed Interactive Simulation for Synthetic Forces," in the *Proceedings of the 11th International Parallel Processing Symposium*, Geneva, Switzerland, April 97

Salomon, B., Govindaraju, N. K., Sud, A., Gayle, R., Lin, M. C., & Manocha, D., “Accelerating Line of Sight Computation Using Graphics Processing Units”, *Proc. of Army Science Conference*, 2004

Wagenbreth, G., Yao, K-T., Davis, D., Lucas, R., & Gottschalk, T., (2005),"Enabling 1,000,000-Entity Simulations on Distributed Linux Clusters," *WSC05-The Winter Simulation Conference,* Orlando, Florida

6.0 List of Presentations

R. Lucas, “Exploiting GPUs for LOS in JSAF”, Graphic Processing Units for Computer Generated Forces project review, UNC, Chapel Hill, NC, Nov. 23, 2004

R. Lucas, “LOS Computations in JSAF”, Graphic Processing Units for Computer Generated Forces project review, UNC, Chapel Hill, NC, May 27, 2005

Appendix A - Abstract submitted to I/ITSEC 2006

Line of Sight and Route Planning Performance using Advanced Architectures

Gene Wagenbreth, Robert F. Lucas
Information Sciences Institute, USC
Marina del Rey, California
{genew, rfl}@isi.edu

ABSTRACT

Current computers usually include a Graphics Processing Unit (GPU). The arithmetic processing capability of these GPUs generally exceeds the capability of the computer's central processing unit (CPU) by an order of magnitude or more. Use of the GPU as an arithmetic accelerator has been discussed by Dinesh Manocha, UNC, and others (IITSEC 2005). The GPU is difficult to program and the calculations to be performed must fit certain criteria in order to use the GPU effectively. This paper examines the feasibility of utilizing these results in the JSAF code in the Urban Resolve experiments. The Joint Semi Autonomous Forces (JSAF) simulation software is used to model hundreds of thousands of entities. Available processing power limits the number of entities simulated on a single CPU. To determine the value of a GPU for JSAF in urban terrain, we looked at two algorithms that utilize a significant portion of the processor capability. These are Line of Sight (LOS) and Route Planning calculations. Both algorithms are contained in a small portion of JSAF source code. This makes translation to GPU code possible. The LOS calculation, particularly when approximated, maps very well onto a GPU. The approximation is such that "can not see" calculations are exact. "can see" calculations must be recalculated exactly on the base CPU. The Urban Resolve trials use terrain dominated by buildings and roads, in contrast to other experiments dominated by natural terrain. In order to determine the feasibility of moving LOS and Route Planning to the GPU, JSAF was instrumented to continuously measure the time spent on these tasks. "can see" and "can not see" results from LOS were separately instrumented. This paper presents the results of running instrumented JSAF in scenarios commonly used by JSAF. A modified LOS approximation algorithm is presented which may allow more efficient execution using the GPU in urban terrain.

ABOUT THE AUTHORS

G**ene Wagenbreth** is a Systems Analyst for Parallel Processing at the Information Sciences Institute at the University of Southern California, doing research in the Computational Sciences Division. Prior positions have included Vice President and Chief Architect of Applied Parallel Research and Lead Programmer at Pacific Sierra Research, where he specialized in tools for distributed and shared memory parallelization of Fortran programs. He has also been active in benchmarking, optimization and porting of software for private industry and government labs. Additional research has been done in scientific programming for porting and optimizing seismic analysis applications. He has programmed on nearly every vector and parallel machine marketed in the US, including CRAY, SGI, Hitachi, Fujitsu, NEC, networked PCs, networked workstations, IBM SP2, as well as conventional machines ranging from PDP11's and VAX's to Pentium IV's. He received a BS in Math/Computer Science from the University of Illinois in 1971.

Robert F. Lucas is the Director of the Computational Sciences Division of the University of Southern California's Information Sciences Institute (ISI). He manages research in large-scale simulations, computer architecture, VLSI, compilers and other software tools. He has been the principal investigator on the JESPP project since its inception in 2002. Prior to joining ISI, he was the Head of the High Performance Computing Research Department for the NERSC at Lawrence Berkeley National Laboratory, the Deputy Director of DARPA's Information Technology Office, a member of the staff, Institute for Defense Analyses, Center for Computing Sciences and Hughes Aircraft. Dr. Lucas received BS, MS, and PhD degrees in Electrical Engineering from Stanford University in 1980, 1983, and 1988.